Communication Project

Group 4

Anh Truong

Nicholas Bui

Shih Tung Yapp

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| 06/22/2022 | 1.0 | Initial Version | Anh Truong, Nicholas Bui, Shih Tung Yapp |
| 7/06/2022 | 1.1 | Added: Use cases A-06 to A-08 and Z-03 | Anh Truong, Nicholas Bui, Shih Tung Yapp |
| 7/18/2022 | 1.2 | Added: Section 2.4, 2.5, 3.1.2, 3.1.3, 3.2, 3.3 and Section 4 | Anh Truong, Nicholas Bui, Shih Tung Yapp |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Purpose 3
   1. Scope 3
   2. Definitions, Acronyms, Abbreviations 4
   3. References 4
   4. Overview 6
2. Overall Description 7
   1. Product Perspective 7
   2. Product Architecture 7
   3. Product Functionality/Features 7
   4. Constraints 7
   5. Assumptions and Dependencies 7
3. Specific Requirements 8
   1. Functional Requirements 8
      1. Common Requirements: 8
      2. Client Module Requirements: 8
      3. Server Module Requirements: 8
      4. Account Module Requirements: 8
   2. External Interface Requirements 8
   3. Internal Interface Requirements 9
4. Non-Functional Requirements 9
   1. Security and Privacy Requirements 9
   2. Environmental Requirements 9
   3. Performance Requirements 9

# Purpose

This document outlines the requirements for the Communication System.

## Scope

This document will catalog the user, system, and hardware requirements for the MPC system. It will not, however, document how these requirements will be implemented.

## Definitions, Acronyms, Abbreviations

* Communication System- Client software to service communication within the company.
* Communications Server- Server to exchange data between users and store logs.
* System User interface: Register, Login, Recover, Delete Account
* Contacts- List of contacts
* Groups- Form groups from contacts to perform group messaging.
* Logs- View message history
* Inbox-  send a private message to another users
* Outbox- Delete the message or try to fix the issue causing the message not to be send.
* Status- A live/or instant updates, to let other users know that this users are on app.
* Read Receipts- messages/notifications showing that “received” users has received the message others sent.
* Search- Search tool used to locate messages in the log with a keyword.
* Delete- Delete messages in the user’s chat log
* Users - Entry level access to standard employees.
* Administrators - Access to system controls to manage users, view logs, and perform system maintenance.
* Use case starting with A indicates user accessible commands while use starting case with Z indicates commands reserve for administrator only.

## References

Use Case ID: A-01

Use Case Name: Users can view messages from others

Requirements: Other users sent messages

Primary Actor: Other users, server

Pre-conditions: Users should be active, and ready to view the message.

Post-Conditions: The other users should send the message, without deleting the messages. The object has been exchanged from the server and sent it back to the user

Basic Flow: 1. The user initiates an action to open the application, and systems responded by showing the messages from other users.

Use Case ID: A-02

Use Case Name: Users can send messages to others

Requirements: Users status was on

Primary Actor: users, server

Pre-conditions: Users should be active, and ready to send the message.

Post-Conditions: The user should send the message, without deleting the messages. The object was exchanged from the server and sent to the other users.

Basic Flow: 1. The user initiates an action to open the application, 2. Then send the message to other users, 3. and systems responded by showing the messages from other users.

Use Case ID: Z-01

Use Case Name: The administrator can create the chat group

Requirements: Admin Role

Primary Actor: Administrators, servers, users.

Pre-conditions: Admin should be active, and ready to create groups. Admin should have an Admin Role.

Post-Conditions: When the administrator has the admin role, he can add users to create a group chat, data can be exchanged from the server, then send notifications to users.

Basic Flow: 1. The administrator initiates an action to open the application, 2. Then add users to create groups, 3. and systems responded by put users in the same group.

Use Case ID: Z-02

Use Case Name: The administrator can remove a user from the chat group the chat group

Requirements: Admin Role

Primary Actor: Administrators, servers, users.

Pre-conditions: Admin should be active. Admin should have an Admin Role. There is a group.

Post-Conditions: When the administrator has the admin role, he can remove users from a group chat, data can be exchanged from the server, then send notifications to users.

Basic Flow: 1. The administrator initiates an action to open the application, 2. Then remove users to create groups, 3. and systems responded by removing users from that group.

Use Case ID: A-03

Use Case Name: The user can receive a notification from the system.

Requirements: User, System

Primary Actor: Servers, users.

Pre-conditions: Other users send a message to the user.

Post-Conditions:  The other users should send the message, without deleting the messages. The object has been exchanged from the server and sent it back to the user. Then the user can receive the message notification.

Basic Flow: 1. The user initiates an action to open the application, and 2. systems responded by showing the messages from other users. 3. User receives the message notification.

Use Case ID: A-04

Use Case Name: The user can delete contacts

Requirements: User, System

Primary Actor: Servers, users.

Pre-conditions: The user has sent some messages before.

Post-Conditions:  The user deletes the contacts

Basic Flow: 1. The user deletes the contacts, and 2. systems responded by removing contacts from users.

Use Case ID: A-05

Use Case Name: The user can search contacts

Requirements: User, System

Primary Actor: Servers, users.

Pre-conditions: The user has sent some messages before.

Post-Conditions:  The user searches the contacts, there will be a list that shows up

Basic Flow: 1. The user deletes the contacts, and 2. systems responded by a list showing up.

Use Case ID: A-06

Use Case Name: The user enter a chat room

Requirements: User, System

Primary Actor: Servers, users.

Pre-conditions: The user have logged into server

Post-Conditions:  User enter the chat room, the server got the signal, then send the broadcast message to other users

Basic Flow: 1. The user logged in, and 2. systems responded and send notification to other users.

Use Case ID: A-07

Use Case Name: The user send message to a chat room

Requirements: User, System

Primary Actor: Servers, users.

Pre-conditions: The user have entered the room

Post-Conditions:  User send a message to the chat room, the server got the signal, then send the user message to other users.

Basic Flow: 1. The user send message in, and 2. systems responded and send that message to other users.

Use Case ID: A-08

Use Case Name: The user leave a chat room

Requirements: User, System

Primary Actor: Servers, users.

Pre-conditions: The user are in the chat room

Post-Conditions:  User leave a the chat room, the server got the signal, then send the notification to other users, and unassigned the user.

Basic Flow: 1. The user leave the room, and 2. systems responded, unassigned the user  and send the notification.

Use Case ID: Z-03

Use Case Name:  Administrator assigned the user to the chat room.

Requirements: User, System

Primary Actor: Servers, users, Admin.

Pre-conditions: The administrator is in the chat room

Post-Conditions:  Admin assigned the user to the chat room, the server got the signal, then send the notification to other users, and assigned the user.

Basic Flow: 1. The administrators assigned user to the chat the room, and 2. systems responded, assigned the user  and send the notification.

For use case and class UML diagram, refer to below.

## Overview

The communication system is design for employees of large organization to chat with each other. As the company would need to supervise employees, administrator possessed the ability to view chat logs of all users.

# Overall Description

## Product Perspective

## Product Architecture

The system will be organized into 3 major modules: the Client module, the Server module and the Account module.

Note: System architecture should follow standard OO design practices..

## Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

## Constraints

2.4.1 Connection cannot be done through local host.

2.4.2 Data cannot be stored in a database.

2.4.3 Chat app cannot be accessed through a web browser.

2.4.4 Chat app only supports text and no other media.

## Assumptions and Dependencies

2.5.1 It is assumed the maximum number of user is less than 100.

2.5.2 It is assumed log are stored in only one file and in plain text.

2.5.3 It is assumed account details in a text file and not encrypted.

# Specific Requirements

## Functional Requirements

### Common Requirements:

3.1.1.1 Every user will have a user id and password.

3.1.1.2 All users can send and receive message.

3.1.1.3 Notification system for unread message.

3.1.1.4 Read receipts for messages.

3.1.1.5 Administrator has the ability to add and remove user from a group.

3.1.1.6 All users has the ability to search logs according to specific keyword.

3.1.1.7 Administrator can read logs of all users.

3.1.1.8 Status information such as online, offline, busy and so on.

3.1.1.9 Users can edit message or delete message within a certain time frame.

3.1.1.10 Users can add or delete contacts.

### Client Module Requirements:

3.1.2.1 Client would need to connect to server unless user choose to exit.

3.1.2.2 Client should only allows user to send and receive messages.

3.1.2.3 Client must be closed if user leaves so that there is no wastage.

### Server Module Requirements:

3.1.3.1 Server would constantly listen and wait for a Client to join.

3.1.3.2 Server would need to able to handle multiple Clients at the same time.

3.1.3.3 Before gaining access to server, users’ login credentials must be verified.

3.1.3.4 Server will not stop listening unless the server is terminated forcefully.

### Account Module Requirements:

3.1.4.1 All accounts would need to have username, password, email, first name, last name, phone number and roles.

3.1.4.2 The two roles are regular user and admin.

3.1.4.3 Password are save in plain text not encrypted.

3.1.4.5 Phone number is a long integer and no special characters are allowed.

## External Interface Requirements

3.2.1 The system has an interface that requires user to login or register to make an account. Account information consists of username, password, email, first name, last name, phone number and role. To obtain an admin account, a special admin code would be required upon registration.

## Internal Interface Requirements

3.3.1 Logs of all users must be stored in a file before the user leaves.

3.3.2 Centralized log files will be updated every time.

3.3.3 All accounts details is stored in a plain text file that is not encrypted.

# Non-Functional Requirements

## Security and Privacy Requirements

4.1.1 Privacy is not required and administrator can view all logs of all users.

4.1.2 Login credentials are required before being able to chat.

4.1.3 Message is not required to be encrypted at any point.

4.1.4 Password is stored in plain text.

## Environmental Requirements

4.2.1 System must be a Java program.

4.2.2 System must be cross platform.

4.2.3 The system require a TCP/IP connection.

4.2.4 System must

## Performance Requirements

4.3.1 Messages needs to be send over from server to client in less than 5 seconds.

4.3.2 Server needs to be able to handle at least 2 clients at the same time.

Team Github Repo: https://github.com/mingzyapp/CS401---Project



